DISCRETE ADJOINT ANALYSIS AND OPTIMIZATION WITH OVERSET GRID MODELLING OF THE COMPRESSIBLE, HIGH-*Re* NAVIER-STOKES EQUATIONS.*

Jonathan Elliott†

Abstract

The development of an "exact reverse" adjoint sensitivity analysis solver based on Chimera overlapping grids, for Euler and turbulent Navier-Stokes physics, and both multigrid and approximate factorization relaxation is presented. The approach results in exact agreement (to within roundoff) at the end of every iteration between the adjoint sensitivity and the sensitivity calculated using a direct method of which the adjoint solver is constructed to be an exact reverse. The adjoint sensitivity analysis methods are incorporated into an optimization system which is shown to reliably optimize a two-dimensional airfoil in viscous, transonic flow and a two-dimensional high-lift configuration in viscous, low-Mach conditions.

Introduction

Adjoint analysis methods are being increasingly used in the fields of optimization (where they can be used to efficiently calculate the gradient and Hessian of the objective) and mesh adaptivity (where they can be used to estimate the nodal contribution to the error in a chosen objective function). For optimizations involving design variables that number more than the number of flow-based objectives and constraints being used, the adjoint method offers the potential for significant CPU savings over the direct method. A capability for direct, discrete sensitivity analysis [11] (that is based on the flow analysis code Overflow) has been extended to perform adjoint analysis.

However the extension did not occur without requiring some problems to be overcome. Previous work for CFD codes with block diagonal solvers, such as point implicit and fourth order Runge Kutta, found that, for these codes, direct (or sensitivity equation) solvers and adjoint solvers could be constructed that have the same asymptotic convergence rates as the flow solver by simply using the same iterative scheme and, for the adjoint solver, merely replacing the direct operations by their transposes [10, 9]. Identical convergence rates were found for all such cases. However, when applying this approach to the block pentadiagonal approximate factorization solver in Overflow [3, 20, 19], it was found that convergence rates for the flow solver were equal to those of the sensitivity solver but not those for the adjoint solver. Although for subcritical flows, adjoint convergence rates were approximately equal, disparities became evident as the freestream Mach number increased. And for some transonic flows, this disparity became unacceptable, requiring significant intervention to obtain convergence. Often the convergence rate was slower than that of the direct and flow solvers, although the final solution was always correct.

To overcome this problem, it was found necessary to use an approach that draws from aspects of the strategy of both the "hand-differentiation" and the "automatic differentiation" camps. The final discretization is similar to that which would be produced by an automatic differentiation process, but incorporates efficiencies such as the storage of certain variables, the bypassing of certain unnecessary calculations and the elimination of the need to store a "log file". In addition, the discretization is restartable. It closely resembles the approach outlined in [12, 13] (which was arrived at independently) but has a simpler, easier to check construction and more straightforward handling of explicit boundary conditions. In addition, the implementation reported herein has been extended to flow solvers modelling the Navier-Stokes equations coupled to a one-equation turbulence model and flow solvers that use Chimera overlapping grids and both multigrid and pentadiagonal approximate factorization relaxation schemes.

Geometry movements were chosen to be simulated by actually moving the grid rather than using transpirations. This choice was made by comparison of the relative disadvantages associated with both. For the target applications of the capability, inaccuracy associated with the transpiration model seemed to outweigh the complications associated with grid movement including treatment of intersections and noise associated with moving grids in a Chimera framework. With regard to the latter issue, probably the more serious of the two, it was felt that, in a similar fashion to the discontinuous response present in shock movement for transonic optimization, this discontinuous behaviour could be at a low enough level that it would not seriously impede the progress of the optimization. This matter was discussed in [11], in which no adverse effect of the discontinuous behaviour had then been observed. How-

*Presented at the 6th Overset Grid and Solution Technology Symposium, Fort Walton Beach, FL,Oct, 2002.

†Principal Engineer, Boeing Commercial Airplanes, Member AIAA

ever more aggressive movement of overset grids in the optimization exercises presented here was expected to provide a challenge to this assumption. As pointed out in [11], recourse can always be made to design variable movements which minimize, or even eliminate changing donor-recipient relationships.

The adjoint implementation is exercised on sample design optimization problems. For single block Navier-Stokes optimizations, typical superlinear optimization convergence behaviour is observed and significant efficiencies over the direct approach are realized. For Chimera optimizations, superlinear convergence is also observed as has been found in previously reported work using the direct approach with overlapping grids [11]. However noise associated with more significantly changing donor/recipient relationships does affect the convergence of the optimization exercises here and user intervention is required to obtain this good convergence. Although there are still issues being dealt with, we find a practical optimization process is certainly emerging.

In the Algorithms section, the components of the optimization system are described. In particular, the development of the chosen adjoint approach is described. The adjoint sensitivities are validated by comparison with values found using the direct method (which in turn have been validated to machine precision level by comparison with central difference approximations for step sizes small enough to reduce truncation error to infinitesimal magnitude but large enough to avoid roundoff error). Finally, the sensitivity analysis scheme is incorporated into an optimization system which is demonstrated to reliably minimize drag at fixed lift and at transonic conditions in two Results sections, one for single-block Navier-Stokes optimizations and one for Chimera optimizations.

Algorithms

Grid generation and perturbation. Baseline grids are generated using HYPGEN [4]. However in the course of a typical optimization, HYPGEN is not used and recourse is made to a simple grid perturbation tool. The modal perturbation method [14, 6] was chosen such that the effect of a design variable, β_i, on the airfoil surface is given by

$$z(x,y) = z_0(x,y) + \sum_{i=1}^{N_{des}} \beta_i sin(\pi(\zeta_i)^{p_i}) \qquad (1)$$

where ζ_i is the normalized axial distance from either the leading or trailing edges. A simple grid perturbation program was written which basically extends these surface movements, for a given value of the streamwise computational variable, into the grid such that Δz is constant for a distance from the surface and then decays linearly to zero at some input value of the normal computational variable. One advantage of this approach is that the analytic grid sensitivities are available practically by inspection.

We have chosen to initially investigate a Chimera grid perturbation approach in which the various grids are allowed to move freely and the interpolation coefficients are allowed to vary. The disadvantage mentioned above is that there could be noise in the functional response to the design variable as a point from one grid moves from a situation in which it lies in one cell in the donor grid to one in which it lies in a different cell, resulting in a discontinous change in the way its interpolation is performed. Also, a point may suddenly become "blanked out" at certain locations of the design space. It is felt, however, if appropriate precautions are taken in the grid construction and in the design variable definition, that this discontinuous behaviour should be at a low enough level that it will not impede the progress of the optimization. Account was taken of the plethora of transonic CFD optimizations which were successful in spite of the discontinuous response present due to shock capturing. The advantage, of course, is that larger grid movements and large changes of design variables are possible without regridding. As discussed later, it has been found that this did not seriously hamper progress of Chimera optimization exercises discussed herein although it did prevent convergence without user intervention, a problem currently being worked.

Flow analysis For flow analysis, we use the Overflow scheme [3], a node-centered algorithm which can be labelled as finite difference or finite volume. For interblock overlap and communication information, we use the Pegsus [23, 1] scheme. This combination of tools is in widespread use in the US aerospace industry [16, 21].

Adjoint sensitivity analysis: "Exact reverse" approach To elaborate on the discussion in the Introduction of adjoint convergence problems, previous work [10, 9] showed that, if the flow solver has a time integration scheme of the form,

$$\mathbf{Q}^{n+1} = \mathbf{Q}^n + \mathbf{L}^{-1}\mathbf{R}(\mathbf{Q}) = \mathbf{Q}^n + \frac{\mathbf{\Delta t}}{\mathbf{\Omega}}\mathbf{R}(\mathbf{Q}) \qquad (2)$$

and adequate convergence of the iteration (here forward Euler) has been achieved, then the direct method time integration scheme

$$\begin{aligned} \mathbf{u}^{n+1} = \frac{d\mathbf{Q}}{d\beta}^{n+1} &= \frac{d\mathbf{Q}}{d\beta}^{n} + \mathbf{L}^{-1}(\frac{\partial \mathbf{R}}{\partial \mathbf{Q}}\frac{d\mathbf{Q}}{d\beta}^{n} + \frac{\partial \mathbf{R}}{\partial \beta}) \\ &= \mathbf{u}^n + \mathbf{L}^{-1}(\mathbf{J}\mathbf{u}^n + \mathbf{f}) \qquad (3) \end{aligned}$$

and the adjoint time integration scheme

$$\bar{\mathbf{f}}^{n+1} = \bar{\mathbf{f}}^n + \mathbf{L}^{-T}\left(\frac{\partial \mathbf{R}}{\partial \mathbf{Q}}^T \bar{\mathbf{f}}^n + \frac{\partial I}{\partial \mathbf{Q}}^T\right)$$

$$= \quad \bar{\mathbf{f}}^n + \mathbf{L}^{-T}(\mathbf{J}^T\bar{\mathbf{f}}^n + \bar{\mathbf{u}}) \tag{4}$$

will both have the same asymptotic convergence rate as that of the flow solver. (Here, shorthand for the Jacobian, $\mathbf{J} = \partial\mathbf{R}/\partial\mathbf{Q}$, the flow sensitivity $\mathbf{u} = d\mathbf{Q}/d\beta$, the residual sensitivity, $\mathbf{f} = \partial\mathbf{R}/\partial\beta$, the cost function, I, the cost function flow sensitivity, $\bar{\mathbf{u}} = \partial I/\partial\mathbf{Q}$ and the adjoint variable, $\bar{\mathbf{f}}$, have been introduced.) Identical convergence rates were found for all such cases. This did not occur for the first implementation of an adjoint solver based on Overflow.

As a means for resolving this problem, recourse was made to the strategy adopted by automatic differentation codes. In this approach [5], the forward method is constructed first, resulting in the differentiation of the entire iterative process. The approach can be approximately thought of as follows. It is assumed that the grid perturbation scheme has already been differentiated (which, upon examination of Equation (1), is not a difficult prospect in some cases), or that the finite difference method has been used, such that the grid sensitivities are available. This vector becomes an additional input to the automatically differentiated code (along with the standard inputs for the flow solver). Thereafter, any quantity which is assigned a value in the flow analysis iterative process, which will have an effect on the ultimate cost function, and which is dependent (even indirectly) on the grid – for example the residual, $\mathbf{R}(\mathbf{Q},\mathbf{X})$ – is differentiated. Then, applying the chain rule, the product of the resulting Jacobian with the input vector is taken. The result is ultimately the cost function sensitivity. Hence the scheme (here applied for three iterations) can be written [5] as

$$\begin{aligned} I_\beta^3\big|_I &= (\bar{\mathbf{u}}, [\mathbf{u}^2 + \mathbf{L}^{-1}(\mathbf{J}\{\mathbf{u}^1 + \mathbf{L}^{-1}(\mathbf{J}\{\mathbf{u}^0 + \mathbf{L}^{-1}(\mathbf{J}\mathbf{u}^0 \\ &\quad +\mathbf{f})\} + \mathbf{f})\} + \mathbf{f})]) \\ &= (\bar{\mathbf{u}}, [(\mathbf{I} + \mathbf{L}^{-1}\mathbf{J})\{(\mathbf{I} + \mathbf{L}^{-1}\mathbf{J})\{(\mathbf{I} + \mathbf{L}^{-1}\mathbf{J})\mathbf{u}^0 \\ &\quad +\mathbf{L}^{-1}\mathbf{f}\} + \mathbf{L}^{-1}\mathbf{f}\} + \mathbf{L}^{-1}\mathbf{f}]) \end{aligned} \tag{5}$$

Here, $I_\beta^n|_I = \bar{\mathbf{u}}^T\mathbf{u}^n$ is the nth iteration estimate of the contribution to the cost function sensitivity due to the flow solution derivative. (The exact total cost function sensitivity

$$I_\beta = I_\beta|_I + I_\beta|_{II} = I_\beta|_I + \partial I/\partial\beta$$

usually has a component that is independent of the flow derivative.) In the forward mode, the sequence of operations in (5) is performed by traversing the expression from the inner set of parenthesis and working outwards. For example, the operations based on the homogeneous term (that dependent on the initial solution) require working from right to left. (Of course, for this linear problem, the final converged sensitivity solution should be independent of the initial condition, $\mathbf{u}^0$. Hence for the homogeneous problem, the only solution dependence – which is transient in nature – is on $\mathbf{u}^0$.) In the reverse or adjoint mode, the sequence of operations is performed in the opposite direction: from outside the outer parenthesis and working inwards. For example, the operations on the homogeneous term require working from left to right. Hence the expressions "forward" and "reverse" modes of differentiation. Since the result of this product is a scalar which is, by construction, independent of the direction of the operations, and since the forward traversal of the product, by inspection, results in the fixed point iteration for the forward method given by Equation (3), this expression suggests the possibility of a fixed point iteration for the adjoint variable which would give precisely the same answer as the forward method. And this would hold for any number of iterations. This would be a valuable tool for debugging since traditional adjoint methods have only had the final result as a check to see whether the coding has been done properly. One tool the developer has resort to is evaluation of terms of the form $(\mathbf{v}_1, \mathbf{A}\mathbf{v}_2) = (\mathbf{A}^T\mathbf{v}_1, \mathbf{v}_2)$, to check individual sections of the code. Determining a fixed point iteration for the adjoint based on Equation (5) allows this idea to be extended to the complete calculation.

Further inspection of Equation (5) reveals that there is one contribution proportional to $\mathbf{u}^0$ and a series of contributions proportional to $\mathbf{f}$. Expansion of the terms in Equation (5) for a series of iterations makes the pattern obvious and, using a rearranged version of Equation (3) for $\mathbf{u}^n$, we can rewrite Equation (5) as

$$I_\beta^n\big|_I \quad = \quad (\bar{\mathbf{u}}, \mathbf{u}^n) = (\bar{\mathbf{u}}, \mathbf{G}^n\mathbf{u}^0 + \mathbf{H}^n\mathbf{f}) \tag{6}$$

where

$$\mathbf{G}^n \quad = \quad (\mathbf{I} + \mathbf{L}^{-1}\mathbf{J})^n \tag{7}$$

$$\mathbf{H}^n \quad = \quad \sum_{i=0}^{n-1}(\mathbf{I} + \mathbf{L}^{-1}\mathbf{J})^i\mathbf{L}^{-1}. \tag{8}$$

If we associate variables with the terms that multiply $\mathbf{u}^0$ and $\mathbf{f}$ to give $I_\beta^n|_I$ (these are the adjoint variables – or the working and adjoint variables, respectively, in [12]), they produce

$$I_\beta^n\big|_I = (\bar{\mathbf{u}}_u^n, \mathbf{u}^0) + (\bar{\mathbf{f}}^n, \mathbf{f}) \tag{9}$$

and comparing with Equations (6)-(8), we must have

$$\begin{aligned} \bar{\mathbf{u}}_u^{n+1} &= \mathbf{G}^{n+1^T}\bar{\mathbf{u}} = (\mathbf{I} + \mathbf{J}^T\mathbf{L}^{-T})^{n+1}\bar{\mathbf{u}} \\ &= (\mathbf{I} + \mathbf{J}^T\mathbf{L}^{-T})\bar{\mathbf{u}}_u^n \end{aligned} \tag{10}$$

$$\begin{aligned} \bar{\mathbf{f}}^{n+1} &= \mathbf{H}^{n+1^T}\bar{\mathbf{u}} = \mathbf{L}^{-T}\sum_{i=0}^{n}(\mathbf{I} + \mathbf{J}^T\mathbf{L}^{-T})^i\bar{\mathbf{u}} \\ &= \mathbf{L}^{-T}\sum_{i=0}^{n}\bar{\mathbf{u}}_u^i = \bar{\mathbf{f}}^n + \mathbf{L}^{-T}\bar{\mathbf{u}}_u^n \end{aligned} \tag{11}$$

with $\bar{\mathbf{u}}_u^0 = \bar{\mathbf{u}}$ and $\bar{\mathbf{f}}^0 = 0$.

Adjoint sensitivity analysis: Explicit boundary conditions. Explicit boundary conditions are simply dealt with using this approach. Equation (5) (for two iterations) becomes

$$I_\beta^2\big|_I = (\bar{\mathbf{u}}, \left[\mathbf{B}\{(\mathbf{I}+\mathbf{L}^{-1}\mathbf{J})\mathbf{B}\{(\mathbf{I}+\mathbf{L}^{-1}\mathbf{J})\mathbf{u}^0 + \mathbf{B}\mathbf{L}^{-1}\mathbf{f}_I\} + \mathbf{f}_B + \mathbf{B}\mathbf{L}^{-1}\mathbf{f}_I\} + \mathbf{f}_B\right]) \quad (12)$$

in which $\mathbf{B} = \partial\mathbf{R}/\partial\mathbf{Q}$ is the Jacobian of the residual for the boundary condition and $\mathbf{f}_B = \partial\mathbf{R}/\partial\beta$. The latter can be present for the inviscid Euler equations, for example, when a design variable has an influence on the normal direction locally resulting in forcing terms of the form $\mathbf{f}_B = \vec{v}\cdot\partial\hat{n}/\partial\beta$. Equation (12) results in the following adjustment to the expressions for the working variable and adjoint variable given in Equations (10)-(11)

$$\bar{\mathbf{u}}_u^{n+1} = (\mathbf{I}+\mathbf{J}^T\mathbf{L}^{-T})\mathbf{B}^T\bar{\mathbf{u}}_u^n \quad (13)$$

$$\bar{\mathbf{f}}_I^{n+1} = \bar{\mathbf{f}}_I^{n+1} + \mathbf{L}^{-T}\mathbf{B}^T\bar{\mathbf{u}}_u^n \quad (14)$$

along with an additional equation for an adjoint variable corresponding to the boundary forcing term

$$\bar{\mathbf{f}}_B^{n+1} = \bar{\mathbf{f}}_B^{n+1} + \bar{\mathbf{u}}_u^n \quad (15)$$

which results in a total estimate of the cost function sensitivity given by

$$I_\beta^N\big|_I = (\bar{\mathbf{u}}_u^{n+1}, \mathbf{u}^0) + (\bar{\mathbf{f}}_I^{n+1}, \mathbf{f}_I) + (\bar{\mathbf{f}}_B^{n+1}, \mathbf{f}_B). \quad (16)$$

Figure 1 shows the evolution of forward and reverse estimates of $I_\beta^n|_I$ for the Euler equations with approximate factorization and multigrid relaxation. For all iterations, agreement is to within thirteen-fourteen significant digits in double precision codes. Note that, here and for adjoint comparisons in subsequent paragraphs, the direct sensitivities have, in turn, undergone thorough validation by comparison with central difference estimates (see [11] for preliminary validation work).

Adjoint sensitivity analysis: Navier-Stokes. This "exact reverse" approach can be easily extended to coupled systems of equations such as the Navier-Stokes equations coupled to a partial differential equation governing a turbulence model. Equation (12) is rewritten – for only one iteration – as

$$I_\beta^1\big|_I = (\bar{\mathbf{u}}, \left[\mathbf{B}_5(\mathbf{I}+\mathbf{L}_5^{-1}\mathbf{J}_5)[\mathbf{B}_6(\mathbf{I}+\mathbf{L}_6^{-1}\mathbf{J}_6)\mathbf{u}^0 + \mathbf{B}_6\mathbf{L}_6^{-1}\mathbf{f}_6] + \mathbf{B}_5\mathbf{L}_5^{-1}\mathbf{f}_5\right]) \quad (17)$$

in which subscript 5 represents terms corresponding to the original five components of the three-dimensional Navier-Stokes equations while subscript 6 represents terms corresponding to the turbulence model. Using the same approach as before, we find new forcing and homogeneous terms corresponding to the turbulence model equation and the following adjusted expressions for those corresponding to the Navier-Stokes equations:

$$\bar{\mathbf{u}}_{u_6}^{n+1} = (\mathbf{I}+\mathbf{J}_6^T\mathbf{L}_6^{-T})\mathbf{B}_6^T(\mathbf{I}+\mathbf{J}_5^T\mathbf{L}_5^{-T})\mathbf{B}_5^T\bar{\mathbf{u}}_{u_6}^n \quad (18)$$

$$\bar{\mathbf{u}}_{u_5}^{n+1} = (\mathbf{I}+\mathbf{J}_5^T\mathbf{L}_5^{-T})\mathbf{B}_5^T\bar{\mathbf{u}}_{u_6}^n \quad (19)$$

$$\bar{\mathbf{f}}_6^{n+1} = \bar{\mathbf{f}}_6^{n+1} + \mathbf{L}_6^{-T}\mathbf{B}_6^T\bar{\mathbf{u}}_{u_5}^{n+1} \quad (20)$$

$$\bar{\mathbf{f}}_5^{n+1} = \bar{\mathbf{f}}_5^{n+1} + \mathbf{L}_5^{-T}\mathbf{B}_5^T\bar{\mathbf{u}}_{u_6}^n \quad (21)$$

which results in a total estimate of the cost function sensitivity given by

$$I_\beta^n\big|_I = (\bar{\mathbf{u}}_{u_5}^n, \mathbf{u}_5^0) + (\bar{\mathbf{u}}_{u_6}^n, \mathbf{u}_6^0) + (\bar{\mathbf{f}}_5^n, \mathbf{f}_5) + (\bar{\mathbf{f}}_6^n, \mathbf{f}_6)$$

Figure 2 shows the evolution of forward and reverse estimates of $I_\beta^n|_I$ for the Navier-Stokes equations (coupled with the Spalart-Allmaras one-equation model) with approximate factorization and multigrid relaxation. For all iterations, agreement is again to within thirteen-fourteen significant digits in double precision codes.

Adjoint sensitivity analysis: Overset grids. Multiblock Chimera systems of equations are also relatively straightforward. For the Overflow implementation applied to a 2-block grid, Equation (5) becomes

$$I_\beta^1\big|_I = (\bar{\mathbf{u}}, \left[(\mathbf{I}+\mathbf{L}_2^{-1}\mathbf{J}_2)\{\mathbf{C}_2[(\mathbf{I}+\mathbf{L}_1^{-1}\mathbf{J}_1)\{\mathbf{C}_1\mathbf{u}^0 + \mathbf{c}_{1s}\} + \mathbf{L}_1^{-1}\mathbf{f}_1] + \mathbf{c}_{2s}\} + \mathbf{L}_2^{-1}\mathbf{f}_2\right]) \quad (22)$$

Here the numerical subscript refers to the block number and it is assumed that, in the flow analysis, the blocks are traversed in order of increasing block number. The matrices $\mathbf{C}_i$ represent interpolation operations in which interpolation information from all other blocks is transferred to the recipient nodes in block i. For example, for a simple combination of two blocks of four nodes each, the operation performed by $\mathbf{C}_i$ in two dimensions is given by

$$\mathbf{C}_i\mathbf{Q} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & a & b & c & d \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \begin{Bmatrix} Q_1^1 \\ Q_2^1 \\ Q_3^1 \\ Q_4^1 \\ Q_1^2 \\ Q_2^2 \\ Q_3^2 \\ Q_4^2 \end{Bmatrix}$$

in which the superscript for an entry in $\mathbf{Q}$ refers to the block in which the node lies. Here the only recipient node in block 1 is node 2 which receives a weighted average of all four nodes in block 2. The scalars a, b, c, d are the weights or interpolation coefficients associated with the four nodes in the donor cell. The vectors $\mathbf{c}_{is}$ refer to source terms that arise when grid movement is such that the interpolation coefficients are a function of the design variables. They are given by (for the simple 2-block two-dimensional example),

$$\mathbf{c}_{is} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & a_\beta & b_\beta & c_\beta & d_\beta \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \begin{Bmatrix} Q_1^1 \\ Q_2^1 \\ Q_3^1 \\ Q_4^1 \\ Q_1^2 \\ Q_2^2 \\ Q_3^2 \\ Q_4^2 \end{Bmatrix}$$

(Note that this formulation for the forward Chimera sensitivity algorithm is a simplification over that reported in [11]. It assumes that the interpolation coefficient sensitivities are available as input. These will have been formed by performing chain rule multiplications

$$\boldsymbol{\xi}_\beta = \boldsymbol{\xi}_{X^d}\mathbf{X}_\beta^d + \boldsymbol{\xi}_{X^r}\mathbf{X}_\beta^r$$

of the matrices representing interpolation sensitivities to donor grid and recipient grid, $\boldsymbol{\xi}_{X^d}$ and $\boldsymbol{\xi}_{X^r}$, respectively with the matrices representing the donor and recipient grid sensitivity $\mathbf{X}_\beta^d$ and $\mathbf{X}_\beta^r$, respectively.) Using the same approach as before, i.e. inspecting (22) and eliciting terms proportional to $\mathbf{u}_i^0$, $\mathbf{f}_i$ and the new forcing terms, $\mathbf{c}_{is}$ corresponding to the Chimera source term, the expressions for the adjoint variables for the Chimera-coupled equations are:

$$\begin{aligned}
\bar{\mathbf{u}}_{u_1}^{n+1} &= \mathbf{C}_1^T(\mathbf{I}+\mathbf{J}_1^T\mathbf{L}_1^{-T})\mathbf{C}_2^T(\mathbf{I}+\mathbf{J}_2^T\mathbf{L}_2^{-T})\bar{\mathbf{u}}_{u_1}^n \\
\bar{\mathbf{u}}_{u_2}^{n+1} &= \mathbf{C}_2^T(\mathbf{I}+\mathbf{J}_2^T\mathbf{L}_2^{-T})\bar{\mathbf{u}}_{u_1}^n \\
\bar{\mathbf{f}}_2^{n+1} &= \bar{\mathbf{f}}_2^{n+1} + \mathbf{L}_2^{-T}\bar{\mathbf{u}}_{u_1}^{n+1} \\
\bar{\mathbf{f}}_1^{n+1} &= \bar{\mathbf{f}}_1^{n+1} + \mathbf{L}_1^{-T}\bar{\mathbf{u}}_{u_2}^n \\
\bar{\mathbf{c}}_{2s}^{n+1} &= \bar{\mathbf{c}}_{2s}^{n+1} + (\mathbf{I}+\mathbf{J}_2^T\mathbf{L}_2^{-T})\bar{\mathbf{u}}_{u_1}^{n+1} \\
\bar{\mathbf{c}}_{1s}^{n+1} &= \bar{\mathbf{c}}_{1s}^{n+1} + (\mathbf{I}+\mathbf{J}_1^T\mathbf{L}_1^{-T})\bar{\mathbf{u}}_{u_2}^n
\end{aligned}$$

which results in a total estimate of the cost function sensitivity given by

$$\begin{aligned}
I_\beta^n\big|_I &= (\bar{\mathbf{u}}_{u_1}^n, \mathbf{u_1}^0) + (\bar{\mathbf{u}}_{u_2}^n, \mathbf{u_2}^0) + (\bar{\mathbf{f}}_1^n, \mathbf{f}_1) + \\
&\quad (\bar{\mathbf{f}}_2^n, \mathbf{f}_2) + (\bar{\mathbf{c}}_{1s}^n, \mathbf{c}_{1s}) + (\bar{\mathbf{c}}_{2s}^n, \mathbf{c}_{2s})
\end{aligned}$$

Figure 3 shows the evolution of forward and reverse estimates of $I_\beta^n|_I$ for the Euler equations – discretized on a two-block grid containing a C-grid and a background box grid. Time integration is, once again, performed using approximate factorization and multigrid relaxation. For all iterations, agreement is again to within thirteen-fourteen significant digits in double precision codes. Figure 4 shows $||\bar{\mathbf{u}}_u^n||_{L_2}$ compared to residuals from the corresponding direct calculations. Note that asymptotic convergence rates for approximate factorization and multigrid are very close to the direct method counterparts. (As shown below, $\bar{\mathbf{u}}_u^n$ represents the residual for the discrete adjoint equation.)

Although these various extensions to the original idea have been introduced in isolation, it is relatively straightforward to implement them together. This has been done for the Overflow flow solver. In addition, the adjoint solver has been extended to three-level multigrid relaxation, in which equality to corresponding forward mode sensitivities is observed on a per-iteration basis to within roundoff. Also matrix and scalar dissipation central difference schemes, flux-difference splitting schemes with Roe's approximate solver, Riemann-/back pressure- as well as characteristic-based farfield boundary conditions have all been incorporated into the adjoint and direct solvers. The differentiation of the flux routines include all the shock-capturing nonlinear devices including pressure-switching and limiters, fully linearized such that locally exact derivatives are found. No failed optimization has been traced to non-differentiable parts of the flux routines to this time.

Adjoint sensitivity analysis: Restartable "exact reverse" approach. Giles [12, 13] independently arrived at the same fixed point iterative scheme for the single block Euler scheme without boundary conditions, barring some minor differences due to different assumed initial conditions for the forward solver. But he went further in that he extended the scheme to a form that can be restarted. (The algorithm given by Equations (10)-(11) is restartable for the same flow solution. But if the solution changes, and sensitivities are sought at a slightly different condition, restarting from previously calculated values of adjoint and working variables will result in an error. This can easily be seen from the fact that, from Equation (11) at convergence, one must have $\bar{\mathbf{u}}_u = 0$. Otherwise, the adjoint variables will be different from the previous iteration. Therefore, when a restart is done using the old variables nothing will change. Everything is driven by $\bar{\mathbf{u}}$. Thus the restart is obviously incorrect. Since $\bar{\mathbf{u}}_u^N$ represents *exactly* the derivative of I_β^N with respect to $\mathbf{u}^0$ and $\bar{\mathbf{f}}^N$ represents *exactly* its derivative with respect to $\mathbf{f}$, when we move to a new flow solution, $\mathbf{Q}$, we have immediately lost this exactness because our current $\bar{\mathbf{f}}$ will have been formed by a series of matrix multiplications found by linearizing a different flow condition - and with a different forcing term. Furthermore, since the solution does not change, we don't even converge to the right answer.)

Giles presented a restartable form (which converges to the right answer even if the flow solution is different than that for which the initial estimates of the adjoint variable were originally calculated) by eliminating the working variable, $\bar{\mathbf{u}}_u$, entirely from the expressions for the adjoint variables. The resulting scheme also matches forward values of the functional on a per-iteration basis. He incorporated inviscid wall boundary conditions by replacing individual rows of the global system for the residuals by ones corresponding to the wall boundary condition. Because of features of the Overflow flow solver including the sequential coupling of flow and turbulence model equations, the sequential coupling of block/block systems and the variety and complexity of boundary conditions, some of which involve updating by extrapolation from the interior followed by averaging of boundary variables, it was found, after learning about Giles' restartable approach, that the "exact reverse" approach given by Equations (10)-(11) was more easily and generally applicable to the Overflow code while still giving per-iteration equality with the forward method.

However to extend the "exact reverse" approach to a restartable form, and carrying along the more general treatment of coupled systems and boundary conditions, it is instructive to follow an approach similar to that presented in [12, 13]. First the iteration is converted to a form allowing discrete integration by parts:-

$$\begin{aligned}\mathbf{u}^{n+1} &= \mathbf{B}\{\mathbf{I}+\mathbf{L}^{-1}\mathbf{J}\}\mathbf{u}^n + \mathbf{B}\mathbf{L}^{-1}\mathbf{f}_I + \mathbf{f}_B \\ &= \mathbf{u}^n + \mathbf{X}\mathbf{u}^n + \mathbf{B}\mathbf{L}^{-1}\mathbf{f}_I + \mathbf{f}_B\end{aligned}$$

where

$$\mathbf{X} = \mathbf{B}\{\mathbf{I}+\mathbf{L}^{-1}\mathbf{J}\} - \mathbf{I}$$

Then, treating the fixed point iteration as a constraint with associated Lagrange multiplier $\bar{\mathbf{u}}_u$, we can augment the functional as follows.

$$\begin{aligned}(\bar{\mathbf{u}},\mathbf{u}^N) &= (\bar{\mathbf{u}},\mathbf{u}^N) - \\ &\quad \sum_{n=0}^{N-1}(\bar{\mathbf{u}}_u^{n+1},\mathbf{u}^{n+1}-\mathbf{X}\mathbf{u}^n-\mathbf{B}\mathbf{L}^{-1}\mathbf{f}_I-\mathbf{f}_B) \\ &= (\bar{\mathbf{u}}-\bar{\mathbf{u}}_u^N,\mathbf{u}^N) + \sum_{n=0}^{N-1}\{(\bar{\mathbf{u}}_u^{n+1}-\bar{\mathbf{u}}_u^n,\mathbf{u}^n)+ \\ &\quad (\mathbf{X}^T\bar{\mathbf{u}}_u^{n+1},\mathbf{u}^n) + (\mathbf{L}^{-T}\mathbf{B}^T\bar{\mathbf{u}}_u^{n+1},\mathbf{f}_I) + \\ &\quad (\bar{\mathbf{u}}_u^{n+1},\mathbf{f}_B)\}\end{aligned}$$

in which discrete integration by parts has been used [12]. Therefore if $\bar{\mathbf{u}}_u$ satisfies the differential equation

$$\bar{\mathbf{u}}_u^n = \bar{\mathbf{u}}_u^{n+1} + \mathbf{X}^T\bar{\mathbf{u}}_u^{n+1} = (\mathbf{I}+\mathbf{J}^T\mathbf{L}^{-T})\mathbf{B}^T\bar{\mathbf{u}}_u^{n+1}$$

with $\bar{\mathbf{u}}_u^N = \bar{\mathbf{u}}$, then the functional is given by

$$(\bar{\mathbf{u}},\mathbf{u}^N) = (\bar{\mathbf{f}}_I^0,\mathbf{f}_I) + (\bar{\mathbf{f}}_B^0,\mathbf{f}_B)$$

where

$$\begin{aligned}\bar{\mathbf{f}}_I^0 &= \sum_{n=0}^{N-1}\mathbf{L}^{-T}\mathbf{B}^T\bar{\mathbf{u}}_u^{n+1} \\ \bar{\mathbf{f}}_B^0 &= \sum_{n=0}^{N-1}\bar{\mathbf{u}}_u^{n+1}\end{aligned}$$

These can be seen to be nearly identical to Equations (13)-(15) found using direct analysis of the product representing the functional given in Equation (12). The only difference is that, in the present case, one marches backwards in time. Following the remainder of Giles' development, with these new expressions,

$$\begin{aligned}\bar{\mathbf{f}}_I^n - \bar{\mathbf{f}}_I^{n+1} &= \mathbf{L}^{-T}\mathbf{B}^T\bar{\mathbf{u}}_u^{n+1} \qquad (23)\\ &= \mathbf{L}^{-T}\mathbf{B}^T(\bar{\mathbf{u}}+\mathbf{J}^T\bar{\mathbf{f}}_I^{n+1}+\mathbf{B}^{*T}\bar{\mathbf{f}}_B^{n+1}) \\ \bar{\mathbf{f}}_B^n - \bar{\mathbf{f}}_B^{n+1} &= (\bar{\mathbf{u}}+\mathbf{J}^T\bar{\mathbf{f}}_I^{n+1}+\mathbf{B}^{*T}\bar{\mathbf{f}}_B^{n+1}) \qquad (24)\end{aligned}$$

in which $\mathbf{B}^{*T} = \mathbf{B} - \mathbf{I}$ has been introduced. Note that $\mathbf{B}^T\mathbf{B}^{*T} = 0$ so, as with Giles' approach, $\bar{\mathbf{f}}_B$ has no effect on $\bar{\mathbf{f}}_I$ and therefore can be calculated in a post-processing step. Also note that the scheme encompasses multiblock Chimera algorithms and one- and two-equation turbulence model systems if, on a per-iteration basis, the interior residuals for all blocks and systems of partial differential equations are calculated and subsequently all the corresponding boundary conditions are applied.

Adjoint sensitivity analysis: Summary. Summing up, the restartable form is obviously preferable. However it has been found to be less amenable to checks of the form $(\mathbf{v}_1,\mathbf{A}\mathbf{v}_2) = (\mathbf{A}^T\mathbf{v}_1,\mathbf{v}_2)$ in which whole segments of the code need to be isolated. As a result, the approach to this point has been firstly to implement the "exact reverse" approach with its simpler, checkable construction and secondly, once this version is in hand, to implement the "restartable exact reverse" version using data from the "checkable exact reverse" version to check the final implementation.

Optimization To drive the optimization process, we have used the Boeing in-house tool named HDNLPR from the BCSOPT library [2] . We used the options that form the Hessian using the BFGS method and used the "make feasible then optimize" algorithm.

Single Block Optimization

Coarse Design Space Discretization. The baseline test geometry is a NACA0012 airfoil. The baseline grid used was that which comes with the standard Overflow distribution [3] and was used to validate Overflow's discretization of various turbulence models against experimental data discussed at the AIAA Viscous Transonic Airfoil Workshop [15]. The baseline freestream flow conditions of $M_\infty = 0.799$, $\alpha = 2.26°$, $Re = 9M$ are identical to one condition used in that experiment. The optimization exercise selected was to minimize C_d at fixed $C_l = 0.346$. The design variable set contained α, the freestream angle of attack, and two shape variables (defined by Equation 1) where $p_1 = 1.3569$, $\zeta_1 = 1-x/c$,

$p_2 = 1.9434$, $\zeta_2 = x/c$ were used resulting in two modes with maxima at $x/c = 40\%$ and $x/c = 70\%$, both with non-zero trailing edge slope. Figure 7 shows the optimal grid. No non-smoothness is observed indicating that the grid perturbation system, described above, is working as planned. Adjoint sensitivities and function values were calculated using the multigrid algorithms and with turbulence simulated using the Spalart-Allmaras model [22] for their respective solvers.

Figures 5-6 shows the evolution of the design variables and objective function, constraint residuals and Kuhn-Tucker residual. Note that the aft camber design variable (x03) has increased while the angle of attack (x01) has decreased. As can be seen from the plots of geometry, pressure coefficient and skin friction shown in Figures 17-19, this has resulted in a highly aft-loaded supercritical airfoil with many of the characteristics expected for an optimal airfoil in this type of flow regime (rapid leading edge expansion, somewhat flat rooftop, etc.) Also note the superlinear convergence evident in the Kuhn-Tucker convergence history which is expected at the end of a quasi-Newton driven optimization with accurate gradients.

The behaviour is similar to that found for a previously reported 2D inviscid optimization [11]. (That case used an equality constraint of Cl=0.493 which is significantly higher than the case here.) For this case, the aft shock movement is limited to a position at around $x/c = 65\%$. At this shock location, the region of separated flow is limited to an area aft of $x/c = 96\%$ on the upper surface as can be seen in Figure 19. The imposition of the pressure distribution found for the optimal inviscid airfoil onto a turbulent boundary layer would most certainly cause much more significant trailing edge separation and result in a far from optimal airfoil.

This is the behaviour expected for viscous optimization and is the main driver for performing viscous rather than inviscid optimization for flow regimes found for (low-Mach) transonic jet transport wings (which rely on airfoils, approximately, of the thickness level and for the flow regimes used here).

Figures 11-12 show a comparison of baseline and optimal C_p distributions while Figures 14-15 show a corresponding Mach comparison. Note that a weak shock has appeared on the lower surface for the optimal solution. The rather high freestream Mach and the constrained thickness distribution (through choice of the design variables) for this case preclude reduction of the drag (and near-elimination of wave drag) to levels found for typical Boeing jet transport airfoils.

Finer Design Space Discretization. With the adjoint solver it is straightforward to increase the fidelity of the optimization to larger numbers of design variables. Although the cost of the optimization can be higher, due to the fact that more major iterations (each involving roughly two adjoint solves and a flow solve) are required till a sufficiently accurate representation of the BFGS Hessian is found, the cost per major iteration does not change significantly. For this exercise, the design space is augmented to include Hicks-Henne modes with peaks at 10%, 20%, 30%, 50%, 60% and 80% in addition to the 40% and 70% modes and α already included in the optimization described in the previous paragraph. Other than that the optimization was defined exactly as that one. Figure 10 shows the optimal grid. Once again, no non-smoothness is observed, indicating an acceptable grid perturbation method, at least for this exercise.

Besides an additional 16 counts of drag improvement, the optimization proceeded as before with about 25 major iterations required till convergence as can be seen in Figures 8-9. Figures 13 and 16 show the optimal C_p and Mach distributions respectively while Figures 17-19 show comparisons of surface quantities superposed on those found in the coarse design space experiment described above. The improved drag is made possible by creation of a sharper leading edge allowing a more rapid expansion, more lift to be carried forward and the aft upper surface shock strength to be decreased. Although improvement is somewhat mitigated by the appearance of a leading edge shock, this shock is weak and doesn't extend far into the flowfield. In addition the shock on the lower surface is weakened. In practical terms, it should be realized that the geometry now represents a very point-designed airfoil. However, its off-design failings could be mitigated by including more flow conditions in the optimization [7].

Note that this adjoint-based optimization realized a savings of about a factor of 4.5 over what the cost would have been had the sensitivities been calculated using the direct sensitivity solver.

Chimera Optimization

2-DV Optimization The baseline geometry for this optimization, shown in Figure 20 is the NLR7301, a two-element landing configuration that can be found in the standard MSES distribution [6] and was used for validating the MSES flow solver since experimental wind tunnel data was available [24]. The geometry was created [8] by taking a rather sensitive, high-thickness, supercritical airfoil and placing a flap in an appropriate location and orientation such that the flow physics found are somewhat representative of what a landing configuration might produce. The wind tunnel test and MSES simulations had a significant stretch of laminar run terminated by a laminar separation bubble. For this optimization, fully turbulent conditions were assumed and simulated with the Spalart-Allmaras model. The baseline flow conditions (near $C_{l_{max}}$) of $M_\infty = 0.4$, $\alpha = 4^\circ$, $Re = 1.5M$ included a significantly higher freestream Mach than would normally be found for flight conditions where the flap is deployed in such a manner. This was done for expediency. The advantage of higher turnaround time far outweighed the benefit of having more accurate simulation since the purpose of the exericise was to test out the optimization system on a high-lift configuration. This situation will change when the low-Mach preconditioning parts of the Overflow code have been incorporated into the adjoint system. Also note that other liberties have been taken with the flow fidelity. Quite dissipative inputs to Overflow have been used. Note that other researchers have recently performed (albeit differently defined) optimizations and validation work using this geometry [17, 18].

The grid (optimal grid is shown in Figure 25) contains four blocks: one C-grid each around the main and flap, and one inner and outer background box grids to allow decreased grid resolution as the farfield is approached. The latter background box grid is not observable from the view shown in Figure 25. The optimization, once again, was chosen to minimize C_d at fixed $C_l = 1.96$. Although this is not the highest priority Navier-Stokes optimization that the high-lift community would like for this configuration (they would like C_l to be maximized), this less problem-prone approach was taken as a first step towards that goal. (Taking this approach avoids having to perform the final stages of the optimization in massively separated conditions with, possibly, associated flow solver convergence problems.)

The design variable set was chosen to be α and flap rotation. Adjoint sensitivities and function values were again calculated using the multigrid algorithms with turbulence modelled using the Spalart-Allmaras model for their respective solvers.

As shown in Figure 23, the objective initially decreases very slightly for the first two iterations as the optimizer drives C_l to the constraint value in the "feasibility" stage. It takes slightly longer, than for simpler cases, to reach feasibility due to the nonlinearity associated with flow features such as the massively separated region. After the end of the second major iteration, the optimizer switches to constrained minimization mode. It quickly drives the objective down from $C_d = 0.0997$ to $C_d = 0.0463$ at the end of the 7th major iteration. At that point the optimizer starts to exhibit non-typical behaviour, with the predicted step size (given by solution of the Newton system for the Kuhn-Tucker residual) having to be repeatedly cut back from. The optimization was stopped and cold-restarted in case this behaviour was due to a poor BFGS estimate for the Hessian. This proved not to be the problem as, in the subsequent restart, the same behaviour was observed in the feasibility stage.

Close examination of the line searches in the second feasibility stage and at the end of first minimization stage revealed that the problem was due to the abovementioned discontinuous behaviour associated with the creation of the new composite grid. The region in design space to which the optimizer is trying to drive the design, in both cases, contains design variable values which result in a composite grid with a different blanking pattern for the inner box grid, than in the region from which the design is marching. The change is unfortunately discontinuous and the point to which the optimizer repeatedly cuts back the step size marks a location adjacent to this discontinuity. There one inner box grid node, close to the flap trailing edge, becomes unblanked. The resulting, sizably discontinuous change in C_l, for example, provides an insurmountable challenge to the models used in the line search algorithm. The workaround used was to use the flap grid from the location before the discontinuity is encountered as a frozen phantom mesh for generating the hole in main and innerbox grids. The subsequent restart shown beginning at tenth iteration in Figure 23 was successful, with feasibility and optimization stages proceeding in the expected fashion. The optimization converged to within tolerance with a final objective of $C_d = 0.0406$.

3-DV Optimization This exercise is identical to that described in the previous paragraph with the exceptions that one additional design variable, a main element camber mode with peak at $x/c = 30\%$, is included and that $C_l = 1.9044$ is the lift constraint.

The optimization, depicted in Figures 26-27, proceeds as before with the exception that the converged objective of $C_d = 0.0316$ was an improvement of about 90 counts beyond that found in the previous exercise. This appears to be mainly due to the complete elimination of the leading edge separation bubble which is effected by drooping of the main element leading edge. The same discontinuous behaviour was observed however at the end of the optimization precluding the development of the usual superlinear Newton convergence tail. The use of a frozen phantom flap mesh proved successful in allowing ultimately convergence of the cold-restarted optimization.

Note that the optimization was slowed down by the appearance of several orphan points in the sixth iteration and one in the ninth through sixteenth iterations and the resultingly inaccurate sensitivities. (The ability to accurately calculate sensitivities in the presence of orphan points is a task that will be accomplished in the near future.) For the seventeenth through twenty-first iterations there are no orphans. The full step suggested by the quadratic Newton model is taken until, at the twenty-first iteration, the step size is cut back because of the same discontinuous behaviour observed in the previous (2-DV) optimization exercise, and discussed above. No orphans appeared after the restart with phantom flap mesh.

Conclusion

A previously reported direct sensitivity solver based on the Overflow code has been extended to perform adjoint sensitivity analysis. This has been performed for both single block and multi-block overlapping grids, for multigrid and approximate factorization relaxation schemes, for Euler, laminar and Spalart-Allmaras Navier-Stokes physics and for both scalar pressure-switched, Roe-averaged matrix dissipation and flux-difference splitting with Roe's approximate solver. The sensitivities have been validated by comparison with direct sensitivities and versus finite difference. The sensitivity analysis scheme has been incorporated into an optimization system which has been demonstrated to reliably minimize drag at fixed lift and at various conditions and for a multielement configuration (although some work remains to be done for the Chimera implementation in order to avoid the requirement for user intervention).

Acknowledgements

Acknowledgement is given to NASA for the technology contributions to several of the components of the optimization system used herein. Thanks are also extended to Venkat Venkatakrishnan, Steve Allmaras, Ernest Roetman, Gordon Blom, Steve Chaney, Bill Huffman, Jeff Slotnick and Dinesh Naik for many helpful suggestions.

References

[1] Benek, J.A., Dougherty F.C. and Buning, P.G. "Chimera: A Grid-Embedding Technique," AEDC-TR-85-64 (AD-A167466), December, 1985.

[2] Betts, J.T., Carter, M.J., Huffman, W.P. "Software for Nonlinear Optimization," Boeing Information and Support Services, MEA-LR-083-R1, June, 1997.

[3] Buning, P.G., Jespersen, D.C., Pulliam, T.H., Klopfer, G.H., Chan W.M., Slotnick J.P., Krist, S.E. and Renze K.J. "Overflow Users Manual Version 1.8j." Technical Report, NASA Ames.

[4] Chan, W.M. and Chiu I. and Buning, P. "User's Manual for the HYPGEN Hyperbolic Grid Generator and the HGUI Graphical User Interface," *NASA TM-108791*, October, 1993

[5] Carle, A., Fagan, M. and Green, L.L. "Preliminary Results from the Application of Automated Adjoint Code Generation to CFL3D," *AIAA-98-4807*, 1998

[6] Drela, M. and Giles, M.B. "Viscous-inviscid analysis of transonic and low Reynolds number airfoils," *AIAA Journal*, V.25(10), 1987.

[7] Drela, M. "Pros and Cons of Airfoil Optimization," in *Frontiers of Computational Fluid Dynamics 1998*, edited by D.A. Caughey, and M.M. Hafez, World Scientific, 1998, pp 363-381.

[8] Drela, M. personal correspondence.

[9] Elliott, J.K., and Peraire, J. "Aerodynamic Optimization using Unstructured Meshes with Viscous Effects," AIAA-97-1849, June, 1997.

[10] Elliott, J.K., "Aerodynamic Optimization based on the Euler and Navier-Stokes equations using Unstructured Grids," PhD Thesis, Dept of Aeronautics and Astronautics, MIT, May, 1998.

[11] Elliott, J.K., and Herling, W. "A Chimera Approach to Aerodynamic Shape Optimization for the Compressible, High-Re Navier-Stokes Equations," AIAA-00-4729, Sept, 2000.

[12] Giles, M. "On the iterative solution of adjoint equations," in *Automatic Differentiation: From Simulation to Optimization* edited by G. Corliss, C. Faure, A. Griewank, L. Hascoet, U. Naumann, Springer-Verlag, 2001.

[13] Giles M., Duta, M.C. and Muller,J.-D., "Adjoint Code Developments Using the Exact Discrete Approach,", AIAA-01-2596, Anaheim, 2001.

[14] Hicks, R.M., and Henne, P.A., "Wing Design by Numerical Optimization", Journal of Aircraft, 15:407-412, 1978.

[15] Holst, T. L. "Viscous Transonic Airfoil Workshop Compendium of Results", Journal of Aircraft, 25:1073-1087, 1988.

[16] Liou, M., Buning, P.G. " Contribution of the Recent AUSM Schemes to the Overflow Code: Implementation and Validation,", AIAA-00-4404, August, 2000.

[17] Nemec, M. and Zingg, D.W., "Multi-point and Multi-objective Aerodynamic Shape Optimization," AIAA-02-5548.

[18] Nielsen, E.J., and Anderson, W.K., "Recent Improvements in Aerodynamic Design Optimization on Unstructured Meshes," *AIAA Journal*, V.40, June, 2002.

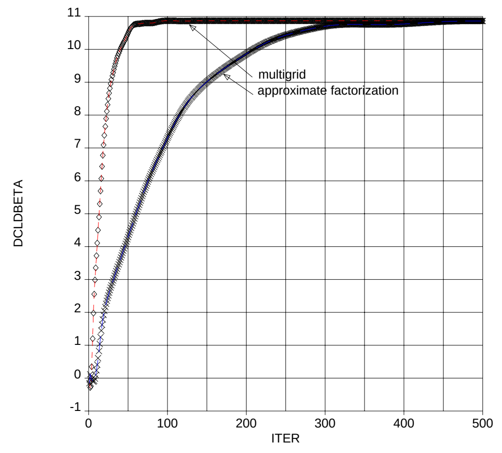

Figure 1: Adjoint and Direct estimates of $I^n_\beta|_I$ for multigrid and approximate factorization relaxation in Euler flow. $M = 0.8$, $\alpha = 2°$. Symbols are direct method; lines are adjoint method.

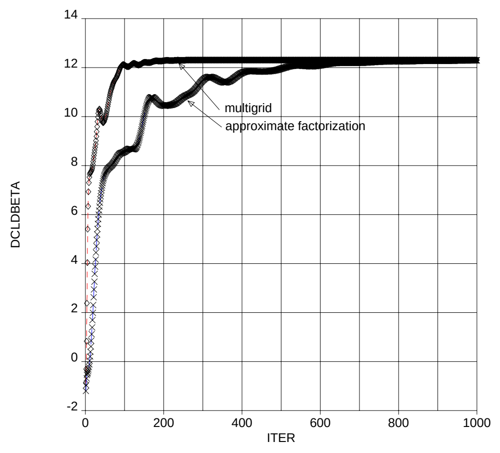

Figure 2: Adjoint and Direct estimates of $I^n_\beta|_I$ for multigrid and approximate factorization relaxation in viscous flow with SA turbulence model. $M = 0.6$, $\alpha = 2°$, $Re = 1.5M$. Symbols are direct method; lines are adjoint method.

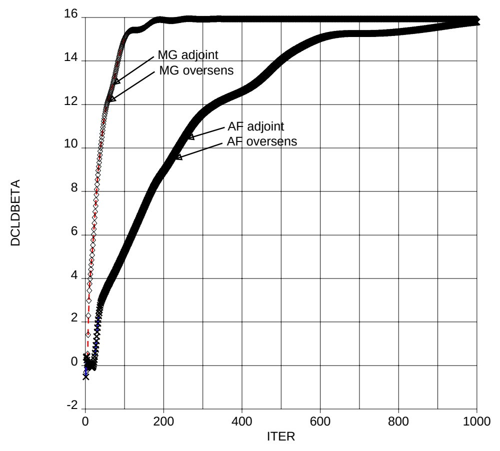

Figure 3: Adjoint and Direct estimates of $I^n_\beta|_I$ for multigrid and approximate factorization relaxation in Euler flow with 2-block Chimera grid. $M = 0.8$, $\alpha = 1.75°$. Symbols are direct method; lines are adjoint method.

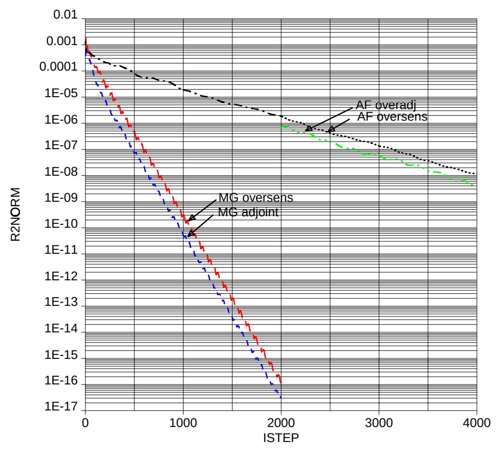

Figure 4: L2 norm residual evolution for multigrid and approximate factorization relaxation in Euler flow with 2-block Chimera grid. $M = 0.8$, $\alpha = 1.75°$

[19] Pulliam, T.H. and Steger, J.L. "Recent Improvements in Efficiency, Accuracy and Convergence for Implicit Approximate Factorization Algorithms," AIAA-85-0360, January, 1985.

[20] Pulliam, T.H. and Chaussee, D.S. "A Diagonal Form of an Implicit Approximate Factorization Algorithm," *Journal of Computational Physics*, V.39, 1981.

[21] Rogers, S.E., Roth, K., Cao, H.V., Slotnick, J.P., Whitlock, M., Nash S.M., Baker, M.D. "Computation of Viscous Flow for a Boeing 777 Aircraft in Landing Configuration," AIAA-00-4221.

[22] Spalart, P. R., and Allmaras, S. R., "A One-Equation Turbulence Model for Aerodynamic Flows," AIAA-92-0439, January 1992.

[23] Suhs, N.E. and Tramel, R.W. "Pegsus 4.0 Users Manual," AEDC-TR-91-8, June, 1991.

[24] van den Berg, B., "Boundary Layer Measurements on a Two-Dimensional Wing with Flap," NLR TR 79009 U, Jan. 1979.

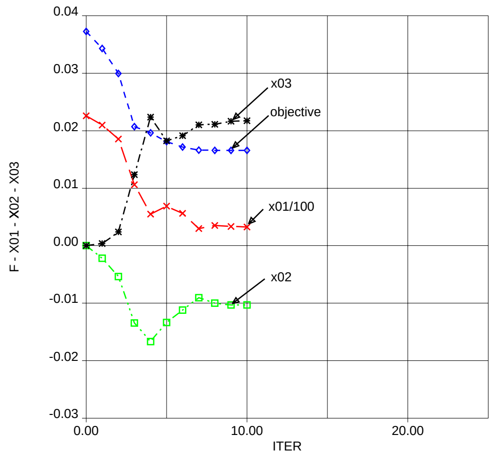

Figure 5: Evolution of objective and DVs in 3-DV single block optimization

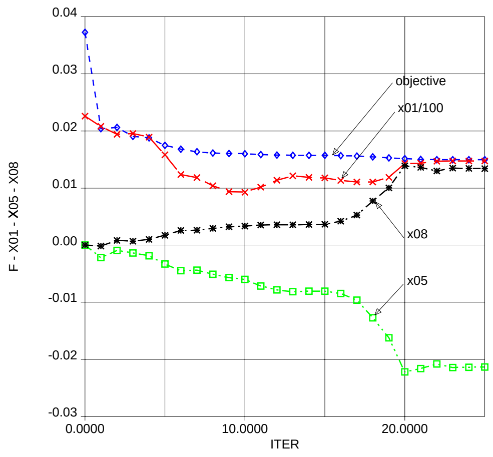

Figure 8: Evolution of objective and DVs in 9-DV single block optimization.

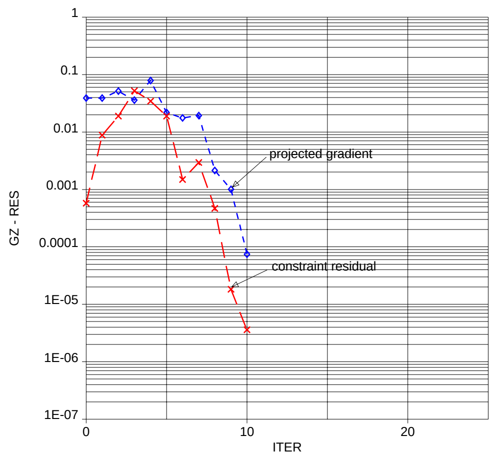

Figure 6: Evolution of Kuhn Tucker and constraint residuals in 3-DV design single block optimization

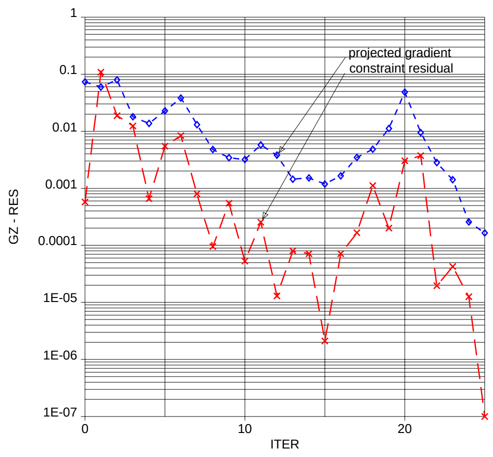

Figure 9: Evolution of Kuhn Tucker and constraint residuals in 9-DV design single block optimization

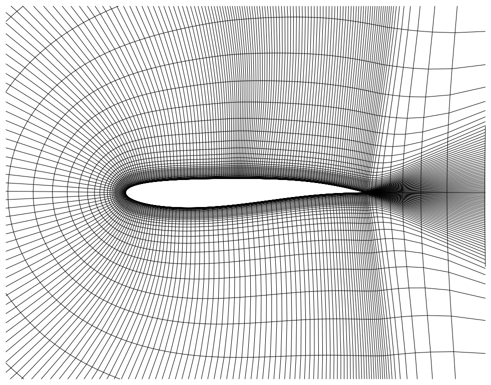

Figure 7: Optimal grid for 3-DV design single block optimization

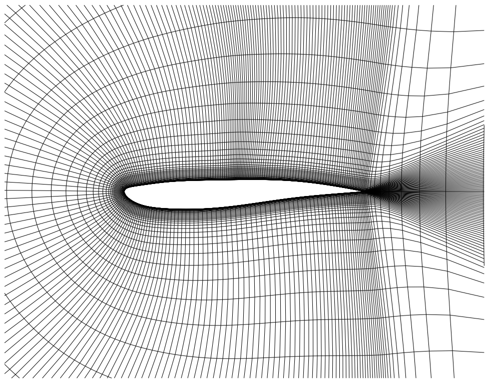

Figure 10: Optimal grid for 9-DV design single block optimization

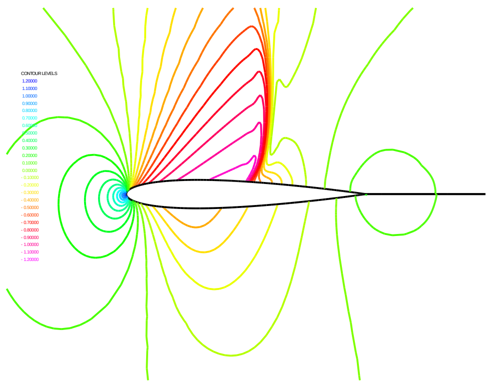

Figure 11: C_p distribution for baseline single block solution.

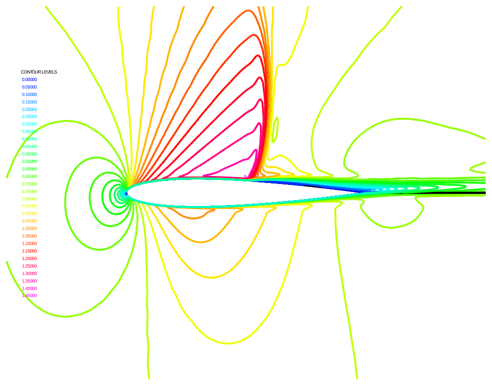

Figure 14: M distribution for baseline single block solution.

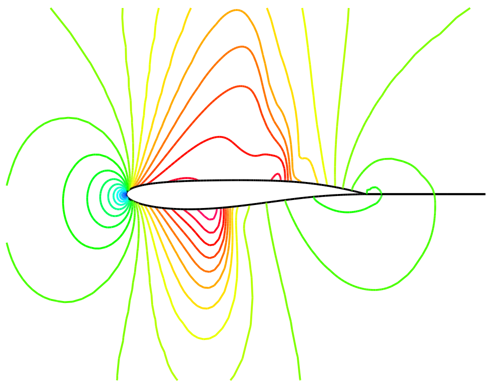

Figure 12: C_p distribution for optimal (3-DV) single block solution.

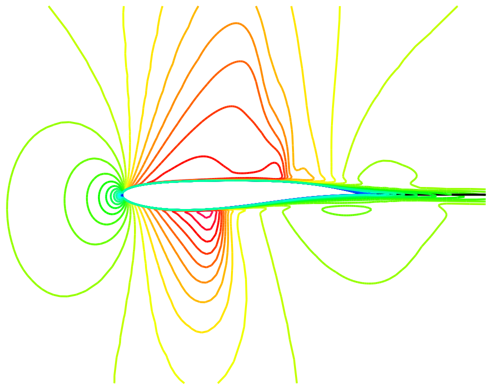

Figure 15: M distribution for optimal (3-DV) single block solution.

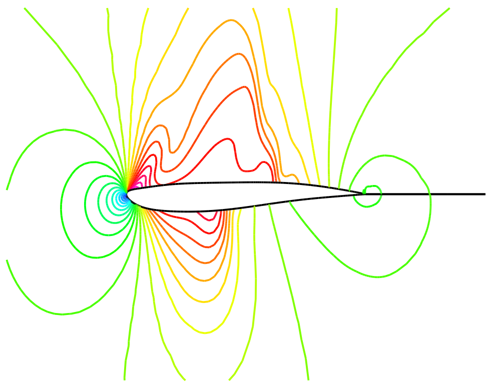

Figure 13: C_p distribution for optimal (9-DV) single block solution.

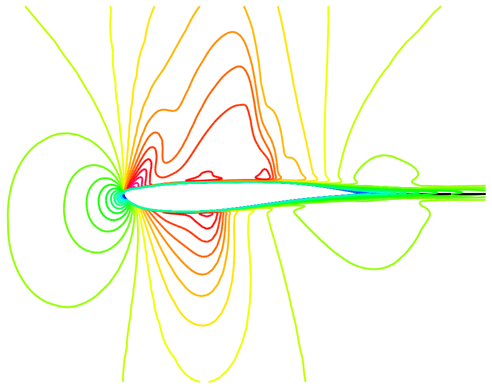

Figure 16: M distribution for optimal (9-DV) single block solution.

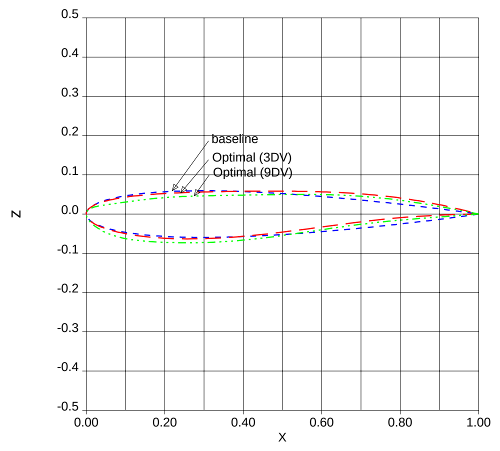

Figure 17: Baseline and optimal geometries for single block optimizations

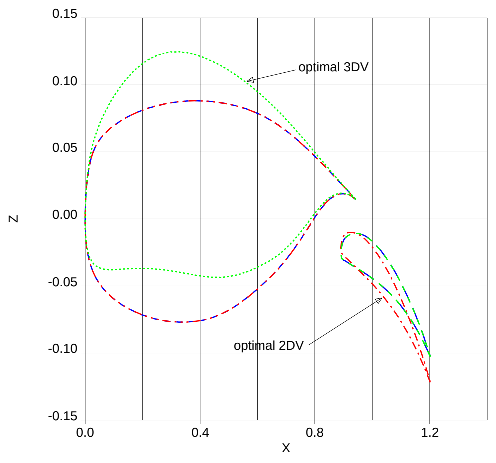

Figure 20: Baseline and optimal geometries for Chimera optimizations

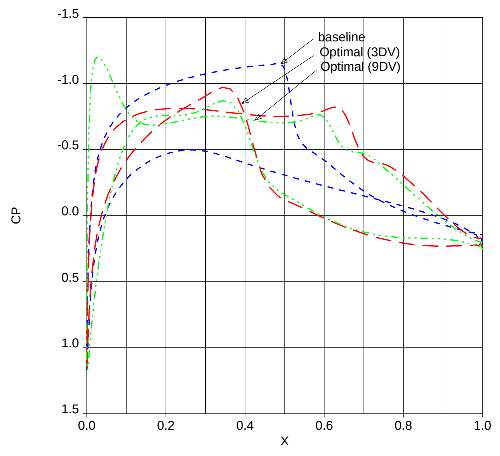

Figure 18: Baseline and optimal surface C_p distributions for single block optimizations

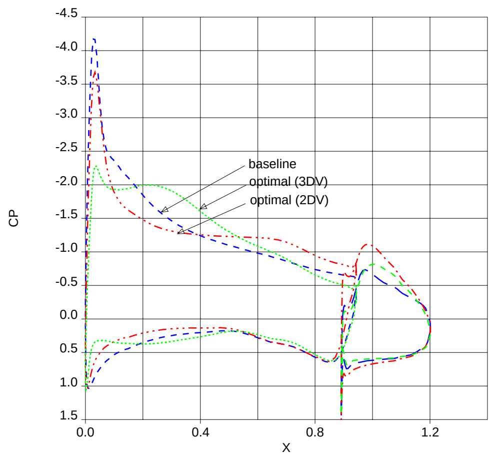

Figure 21: Baseline and optimal surface C_p distributions for Chimera optimizations

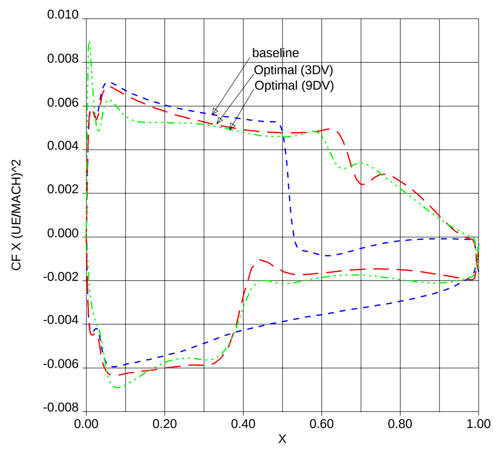

Figure 19: Baseline and optimal surface C_f distributions for single block optimizations

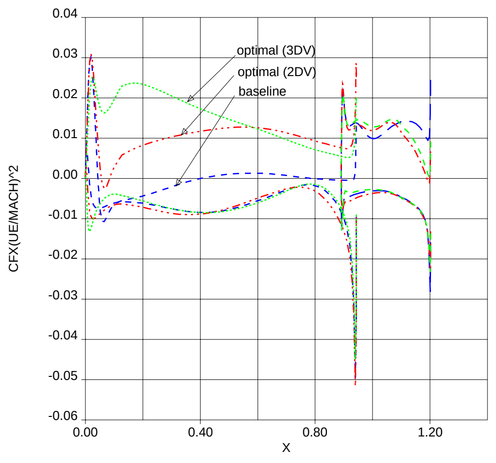

Figure 22: Baseline and optimal surface C_f distributions for Chimera optimizations

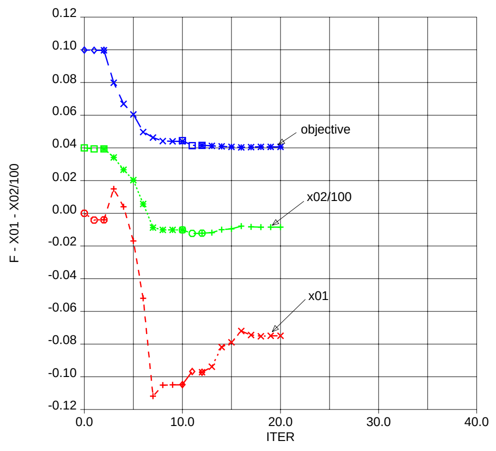

Figure 23: Evolution of objective and design variables in 2-DV Chimera optimization

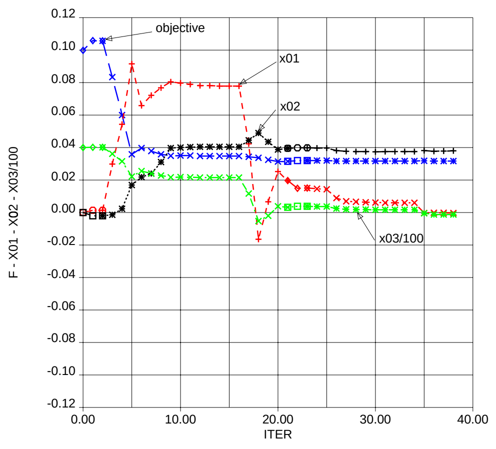

Figure 26: Evolution of objective and design variables in 3-DV Chimera optimization.

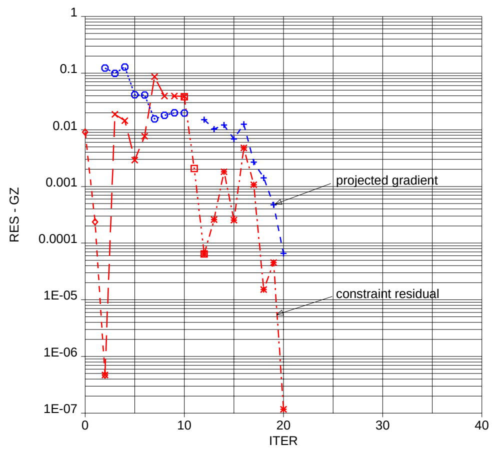

Figure 24: Evolution of Kuhn Tucker and constraint residuals in 2-DV Chimera optimization

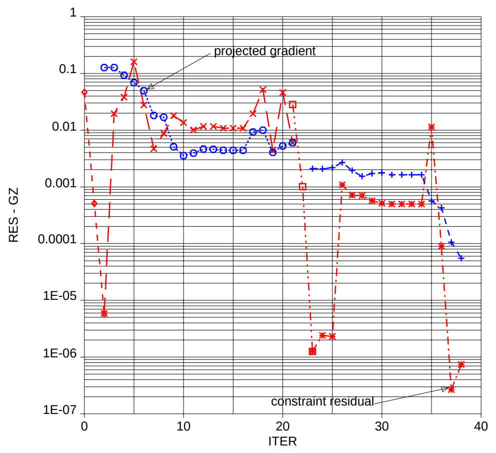

Figure 27: Evolution of Kuhn Tucker and constraint residuals in 3-DV Chimera optimization

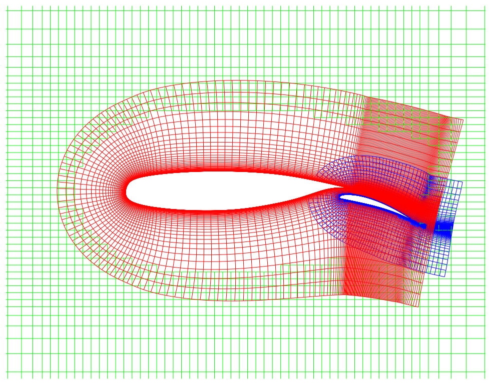

Figure 25: Optimal grid for 2-DV Chimera optimization

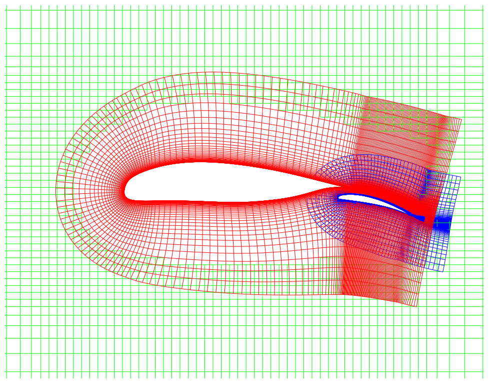

Figure 28: Optimal grid for 3-DV Chimera optimization

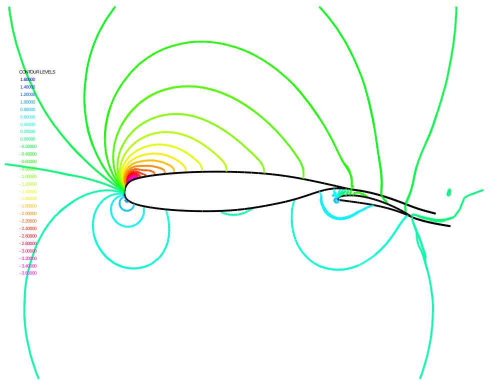
Figure 29: C_p distribution for baseline Chimera solution.

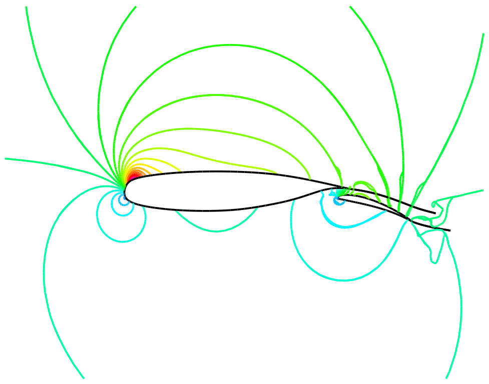
Figure 30: C_p distribution for optimal (2-DV) Chimera solution.

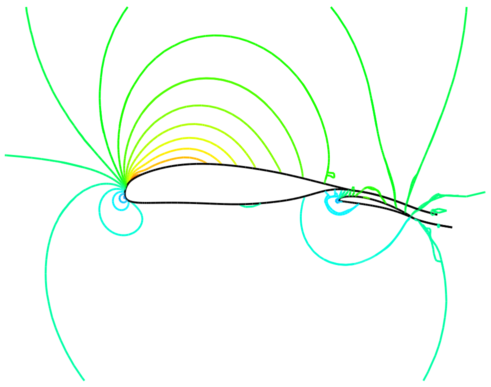
Figure 31: C_p distribution for optimal (3-DV) Chimera solution.

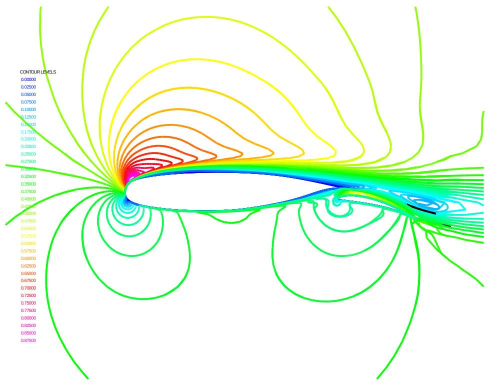
Figure 32: M distribution for baseline Chimera solution.

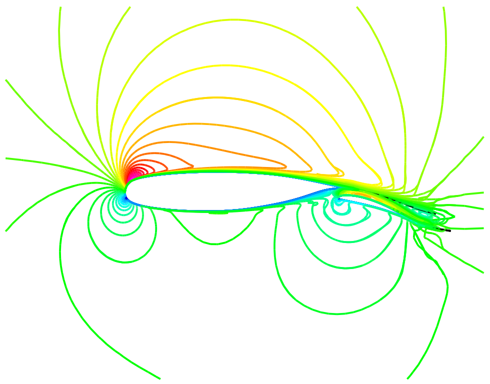
Figure 33: M distribution for optimal (2-DV) Chimera solution.

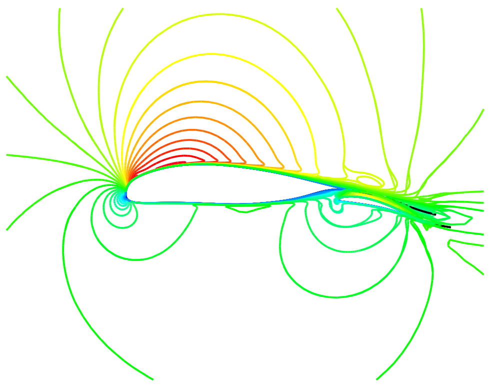
Figure 34: M distribution for optimal (3-DV) Chimera solution.